Remarks/Arguments

Applicants respectfully request consideration of the subject application as amended herein. This Amendment is submitted in response to the Final Office Action mailed July 28, 2008. Claims 1-45 are rejected.

In this Amendment, claims 43 and 44 have been amended. No claims have been added or cancelled. It is respectfully submitted that the amendments do not add new matter.

Applicants reserve all rights with respect to the applicability of the Doctrine of Equivalents.

Claim Rejections under 35 U.S.C. § 112

The Examiner has rejected claims 43 and 44 under 35 U.S.C. § 112, second paragraph, as being indefinite. In particular the Examiner notes that "Java Virtual Machine" is a trade name, thus rendering the claims indefinite (Final Office Action, mailed 7/18/08, page 4). Applicants have accordingly amended claims 43 and 44 to remove the references to "Java Virtual Machine." In light of the amendments, Applicants respectfully request withdrawal of the rejection.

Claim Rejections under 35 U.S.C. §103(a)

Claims 1-12, 15, 17-20, 21-32, 35, 37-41, 43, and 44 were rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Publication No. 2002/0129060 of Rollins et al. (hereinafter "Rollins") in view of U.S. Patent No. 6,675,354 of Claussen et al. (hereinafter "Claussen"). Applicant respectfully disagrees.

Rollins describes a system and method for generating multiple customizable interfaces for XML documents (Rollins, Abstract). A component generation engine uses XML schema, which defines the structure of an XML document, and optional user preferences to define multiple components (Rollins, page 2, paragraph 0031; page 3, paragraphs 36-37). Then, a code-generator generates "a set of Java classes designed to mediate communication between the user and the synchronized tree manager" (Rollins, page 3, paragraph 0038).

Claussen describes processing and handling custom tags in a document (Claussen, Column 3, lines 14-29). The custom tags may be processed regardless of the type of tag that is encountered (Claussen, Column 30, lines 30-42). To avoid errors in processing the various tags, a tag preprocessing technique is utilized to classify the tag according to a tag library (Claussen, Column 6, lines 8-32). However, Claussen merely describes routines for translating various custom tags and handling the tags according to their library definition.

Claim 1 recites:

> In a computer system, an improved method for developing a Web application, the method comprising:
>
> providing a Web application development framework, said framework including an abstract command tag that predefines at least some generic Web application activities;
>
> specifying at least one custom action that is desired to be performed by a Web application;
>
> creating an object-oriented programming language (OOPL) class that extends the abstract command tag for providing execution logic for said at least one custom action, in addition to pre-existing logic that supports said at least some generic Web application activities, thereby creating a corresponding customized command tag that is capable of being embedded within a Web page;
>
> embedding the customized command tag in a Web page of the Web application; and
>
> upon execution of the Web application including an embedded customized command tag in a Web page, invoking the customized command tag for conditionally executing said specified at least one custom action based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag.

(Emphasis Added)

Applicant respectfully submits that a combination of Rollins and Claussen fails to teach or suggest "upon execution of the Web application including an embedded customized command tag in a Web page, invoking the customized command tag for conditionally executing said specified at least one custom action based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag."

Rollins describes a component generation engine that interprets an XML document based upon user preferences and various modes selected for the XML document. Although Rollins briefly mentions that web programming languages utilize tags (*See* Rollins, paragraphs [0004-0010]), Rollins is silent as to the embedding and conditional execution of customized tags in a web application, as recited in claim 1 (Final Office Action, 07/18/08, page 6).

The Examiner therefore relies on Claussen and states that:

> Claussen discloses a document object model (DOM) tree is processed to identify custom tags. Upon encountering a custom tag, an appropriate tag handler (e.g., a Java objet, an XSL stylesheet, or the like) is invoked. A tag registration routing is used for recognizing and handling case-insensitive tags. A servlet engine is examining a tag name, if the name does not match one of the registered tags, the routine converts the name to lower or neutral case. If the tag recognition routing recognizes the name as one of the case-insensitive tags, it converts the attributes to the appropriate case, and hand the resulting element off to a correct tag handler for processing (col. 3, lines 30-44; col. 7, lines 5-35). Here Claussen is clear that custom tags are handled by appropriate tag handler i.e., invoking appropriate tag handler based upon custom tag encountered, i.e., custom tag are being processed conditionally.
>
> (Final Office Action, 07/18/08, page 3 [Emphasis Added])

Applicant respectfully disagrees.

As discussed in Claussen and as noted by the Examiner, a data type definition (DTD) includes a class or stylesheet attribute in the class definition for each tag (Final Office Action, 07/18/08, page 3 *citing* Claussen at col. 3, lines 30-44; col. 7, lines 5-35). The class or stylesheet attribute in the definition "dictate which of the two mechanisms are used to handle custom tags" (Claussen, column 8, lines 1-3). Based on a tag's registration, the tag is handled (i.e., executed) by one of the two appropriate tag handlers. Thus, in Claussen, the actions associated with tags are always carried out by one of two tag handling mechanism. There is no hint or suggestion, however, that Claussen teaches or suggests conditional execution of any action. That is, since Claussen describes handling each tag by one of two mechanisms, Claussen fails to teach that a tag may fail to be executed at all.

The Examiner states, however, that "Claussen is clear that custom tags are handled by appropriate tag handler i.e., invoking appropriate tag handler based upon custom tag encountered, i.e., custom tag are being processed conditionally" (Final Office Action, 07/18/08, page 3). Applicants respectfully submit that they are unclear how handling all tags by one of two appropriate tag handler based on a tag type, could teach or suggest conditionally executing a customized action specified in a tag. The tag handling mechanisms of Claussen merely determine how to execute tags, and not whether or not to execute a tag.

Claim 1 recites "upon execution of the Web application including an embedded customized command tag in a Web page, invoking the customized command tag for conditionally executing said specified at least one custom action based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag." That is, in accordance with the language of claim 1, the custom action is conditionally executed based on runtime conditions of the Web application and attributes in the tag. Because Claussen teaches deciding which of two handling mechanisms to apply to each tag in a document before executing tags, Claussen fails describe any conditional execution of tags based on status of a Web application or on the attributes in a tag.

Thus, a combination of Rollins and Claussen, whether taken alone or in combination, fail to teach or suggest "upon execution of the Web application including an embedded customized command tag in a Web page, invoking the customized command tag for conditionally executing said specified at least one custom action based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag." Therefore, Applicant respectfully submits that claim 1, and the claims that depend on it, are not rendered obvious by Rollins and Claussen.

Claim 21 recites:

> A computer-readable storage medium storing instructions for a Web application framework, which when executed by a computer system, causes the computer system to perform a method comprising:
>
> specifying an abstract command tag that predefines at least some generic Web application activities;
>
> providing a programming environment for:

(i) specifying at least one custom action that is desired to be performed by a Web application under development, by supporting creation of an object-oriented programming language (OOPL) class that extends the abstract command tag for providing execution logic for said at least one custom action, thereby creating a corresponding customized command tag that is capable of being embedded within a Web page, wherein said customized command tag includes the ability to conditionally execute said specified at least one custom action based on run-time conditions; and

(ii) enabling embedding of the customized command tag in a Web page of the Web application;

wherein execution of the Web application includes invoking the customized command tag for conditionally executing said specified at least one custom action based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag.

(Emphasis Added)

As discussed above, with respect to claim 1, Rollins and Claussen fail to teach or suggest embedding and conditional executing customized tags in a web application based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag. Claim 21 as amended recites "(ii) enabling embedding of the customized command tag in a Web page of the Web application; wherein execution of the Web application includes invoking the customized command tag for conditionally executing said specified at least one custom action based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag." Thus, claim 21, and the claims that depend form it, are not rendered obvious over Rollins in view of Claussen.

Claim 41, as amended, recites:

An improved method for Web application development, the method comprising:

providing a Web-based application development framework built from a set of object-oriented programming language (OOPL) classes, which extends an abstract command tag, said framework providing:

a non-programmatic tag framework that implements the functionality of the application framework when executing within a Web page using dynamic scripting capability;

tag-based Web application objects controlling program flow, executing user commands, representing application business objects, and constructing output;

a non-programmatic tag framework that accesses data for logical business objects and allows page designers to specify an action to be performed;

enabling the embedding of the tag-based Web application objects in a Web page of a Web application; and

execution of the Web application including invoking the tag-based Web application objects for conditionally executing actions specified by page designers based on run-time conditions of the tag-based Web application and tag attributes in the tag-based Web application objects.

(Emphasis Added)

As discussed above, with respect to claims 1 and 21, Rollins and Claussen fail to teach or suggest embedding and conditional executing customized tags in a web application based on run-time conditions of the Web application and run-time values for one or more attributes included in the customized command tag. Claim as amended 41 recites "enabling the embedding of the tag-based Web application objects in a Web page of a Web application; and execution of the Web application including invoking the tag-based Web application objects for conditionally executing actions specified by page designers based on run-time conditions of the tag-based Web application and tag attributes in the tag-based Web application objects." Thus, claim 41, and the claims that depend form it, are not rendered obvious over Rollins in view of Claussen.

Claims 13, 14, 16, 33, 34, 36, 42, and 45 were rejected as being unpatentable over Rollins and Claussen in view of U.S. Patent No. 6,760,748 of Hakim (hereinafter "Hakim"). Applicants respectfully disagree.

As discussed above Rollins and Claussen fail to describe each and every limitation claimed by the Applicant. Hakim merely describes an electronic classroom in which data is transmitted from a teacher's terminal to student terminals (Hakim, Column 3, lines 15-21; Figure 2). Hakim does not address the use of customized tags in a Web application. Therefore, Hakim fails to remedy the shortcomings of Rollins and Claussen discussed above. Thus, Rollins, Claussen, and Hakim, alone or in combination, fail to render claims 1, 21, and 41, and thus dependent claims 13, 14, 16, 33, 34, 36, 42, and 45, obvious.

Claims 14, 16, 34, 36, and 45 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Rollins and Claussen in view of U.S. Patent No. 6,605,120 of Fields et al. (hereinafter "Fields"). Applicants respectfully disagree.

As discussed above Rollins and Claussen fail to describe each and every limitation claimed by the Applicant. Fields describes a web page filter that extracts content from a web page based on filter definitions (Fields, column 5, liens 12-34). Fields does not address the use of customized tags in a Web application. Therefore, Fields fails to remedy the shortcomings of Rollins and Claussen discussed above. Thus, Rollins, Claussen, and Fields, alone or in combination, fail to render claims 1, 21, and 41, and thus dependent claims 14, 16, 34, 36, and 45, obvious.

Conclusion

Applicant respectfully submits that in view of the amendments and discussion set forth herein, the applicable rejections have been overcome. Accordingly, the present and amended claims should be found to be in condition for allowance.

If a telephone interview would expedite the prosecution of this application, the Examiner is invited to contact Judith A. Szepesi at (408) 720-8300.

If there are any additional charges/credits, please charge/credit our deposit account no. 02-2666.

Respectfully submitted,
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: September 18, 2008

/Judith Szepesi/
Judith A. Szepesi
Reg. No. 39,393

Customer No. 08791
1279 Oakmead Parkway
Sunnyvale, CA 94085
(408) 720-8300